The Hidden Footprint: Making Storage a First-Class Metric for LLM Agent Evaluation

Chenglin Yu[1], **Hongquan Gui**[1], **Ying Yu**[2], **Hongxia Yang**[3], **Ming Li**[1,4*]
[1]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University
[2]College of Economics and Management, Zhejiang Normal University
[3]Department of Computing, The Hong Kong Polytechnic University
[4]Research Institute for Generative AI, The Hong Kong Polytechnic University
chenglin.yu@polyu.edu.hk, ming.li@polyu.edu.hk

Abstract

LLM agent benchmarks measure task completion, reliability, and inference cost, but not the persistent data an agent run leaves on disk—logs, context snapshots, checkpoints, debug traces. These bytes are absent from the leaderboards we survey, yet bear on whether an agent system can be deployed on desktops, under data-retention compliance regimes, or at fleet scale. We introduce AgentFootprint, to our knowledge the first systematic cross-framework benchmark of post-run agent storage footprint. Its serialization-aware metric suite—total retention, channel composition, duplication, growth exponent, compressibility, and a conversation-history reconstructability score—addresses a measurement trap: naïve byte-level measurement understates duplication by an order of magnitude, because database paging and JSON escaping obscure repeated content. The footprint has two determinants—the logical volume the agent's behavior generates, and the amplification the persistence layer adds when storing it—and a fixed-trace control isolates the second: the same trajectory replayed through each persisting framework's adapter yields retained sizes differing by $6.7\times$. Under identical models, tools, and tasks, framework configurations at identical 100% accuracy differ by $15.7\times$ in retained bytes; the defaults bundle different recovery and audit capabilities, so the suite reports storage jointly with accuracy and reconstructability. Three full-history configurations grow superlinearly on a repeated-observation stress task, and on a deliberately minimal write task framework residue exceeds the delivered output files by orders of magnitude. Exported trajectories from 108 instance-normalized SWE-bench Verified submissions span three orders of magnitude in per-instance volume with no detectable correlation with resolve rate. A content-addressed store reduces retention 4.8–$32.7\times$ while preserving all properties checked by our restoration tests, including every reconstructability score: exact history reconstruction does not require megabytes, and these storage metrics can be reported alongside inference cost.

1 Introduction

LLM agents—systems that pursue multi-step tasks by interleaving model calls with tool use—are moving from demos into production, and the community evaluates them along an expanding set of axes: task success on GAIA (Mialon et al. 2024) and SWE-bench (Jimenez et al. 2024), reliability via pass^k on τ-bench (Yao et al. 2025), and, following Kapoor et al. (2025), inference cost. A recent survey of agent evaluation catalogs more than a dozen such metrics (Mohammadi et al. 2025)—none of them includes the *storage footprint*: the bytes an agent execution leaves behind on disk. This paper measures that axis.

Unlike inference cost, which is consumed when execution ends, retention is a stock that persists and *accumulates* across runs. This persistence is simultaneously useful and costly: it is the substrate for replay debugging, recovery, and compliance audits, and it enlarges long-term storage, governance, and deletion obligations. The magnitudes involved are easy to underestimate. In a production deployment that we measured directly—a supply-chain quoting system (Supp. App. P)—a single task retained 131 MB, including 79 MB of framework residue, compared with 2.6 MB of delivered artifacts; per-task retention had a median of 715 KB and a 90th percentile of 81 MB. The deployment as a whole has accumulated $\sim$56 GB over $\sim 10^3$ recorded rounds (operator-attested). For illustration, at the same per-round average a fleet processing 10^4 rounds/day would accumulate $\sim$560 GB daily if nothing is deleted. Storage is to agents what memory footprint was to mobile apps—a resource that went unmeasured until it constrained deployment.

Our goal is to make this axis measurable and comparable across agent frameworks. Doing so is challenging for three reasons. First, *attribution*: an agent run scatters bytes across working directories, session databases, hidden home-directory state, and tokenizer caches; deciding which bytes are task-attributable requires per-run isolation, not a disk-usage command. Second, *serialization masking*: the same file content is re-stored many times inside SQLite pages and JSON-escaped message arrays, where page fragmentation and escape sequences defeat byte-level fingerprinting—reporting $D{=}1.01$ for a store logically holding the same content twelve times over. Third, *semantics*: raw byte counts alone cannot distinguish an agent that retains much to support auditing from one that retains much because of avoidable storage amplification; the metric suite must indicate which recovery and audit capabilities the retained bytes support.

We present AgentFootprint, a benchmark and measurement harness addressing all three. Each (framework, task) pair runs in a fresh sandbox whose workspace and home delta is captured and attributed (challenge one). Our meter analyzes *logical content streams*—SQLite cells, JSONL

*Corresponding author.

records—rather than raw files, and injects content probes that are matched under raw and JSON-escaped encodings (challenge two). On top of this substrate we define six measurements: total retention S_{total}, a descriptive storage-channel composition, duplication factor D, growth exponent α fitted over controllable-horizon tasks, long-window compressibility C, and a 0–3 history-reconstructability score R asking whether retained bytes reconstruct the conversation history behind step k (challenge three); a four-way semantic taxonomy aids interpretation. Throughout, storage is a system-level *resource* dimension, read jointly with accuracy and R, never a capability by itself; its two determinants are separated by a fixed-trace control (§5.6).

Measuring eight representative frameworks—LangGraph, AutoGen, CrewAI, SmolAgents, OpenAI Agents SDK, LlamaIndex, Agno, and InfiAgent—under identical backends, tools, and tasks yields three findings. (i) *Equal-accuracy dispersion*: at identical 100% task accuracy, retained bytes differ by $15.7\times$ (LangGraph vs. CrewAI), and one framework retains nothing at all and accordingly can recover or resume nothing. (ii) *Growth regimes*: frameworks that persist full history at every step exhibit α up to 1.95—a 200-round monitoring loop over one 2 KB status file leaves 323 MB—while frameworks with windowed context management stay near-linear. (iii) *Residue dominates artifacts*: when agents produce legitimate output files, framework residue exceeds those artifacts by up to $24{,}033\times$.

Public agent submissions exhibit an analogous dispersion in exported trajectory volume: across 108 instance-normalized SWE-bench Verified submissions, per-instance volume spans $1{,}617\times$ with no detectable correlation with resolve rate—in this snapshot, heavier trajectories are not more successful ones. Nor is the footprint required for reconstructable history: a content-addressed store removes most retained bytes with all restore checks passing and every R score reproduced.

Our contributions are:

- **A metric suite and measurement methodology** for agent storage footprint (S_{total}, composition, D, α, C, R), including the finding that serialization masks duplication from naïve measurement by up to $12\times$ (§3).
- **AgentFootprint**, a reproducible harness and task suite measuring eight frameworks over 1,061 sandboxed runs—a 509-run main study (three repetitions plus ablation, growth, write, exploratory, heterogeneous, and fixed-trace suites) plus 552 replication and variant runs (two further backends, growth seeds, executed history-only forms, shared threads, replay-source tasks)—with container-validated isolation (§4–§5; full accounting in Supp. App. C).
- **An in-the-wild study** harvesting all 134 SWE-bench Verified submissions (108 instance-normalized), finding a $1{,}617\times$ volume spread with no detectable correlation between exported volume and success ($\tau_b = 0.080$, $p = 0.222$) (§6).
- **A headroom demonstration**: an existence proof, not a proposed storage system—a content-addressed compactor showing the measured reconstructability score survives at 4.8–$32.7\times$ fewer bytes than the frameworks' defaults (§7).

2 Related Work

AgentFootprint belongs to a line of work that expands *what* agent benchmarks measure rather than *how well* agents score. Kapoor et al. (2025) introduced inference cost as a mandatory reporting axis and showed simple baselines dominate the cost–accuracy Pareto frontier; τ-bench added reliability via pass^k (Yao et al. 2025). Storage footprint is orthogonal to both and differs in kind: cost is a flow that stops with the task; retention is a stock that persists, compounds, and carries compliance and reproducibility obligations. Efficient Agents studies effectiveness against inference cost (Wang et al. 2025); MAFBench reports architecture-dependent latency and coordination costs (Orogat, Rostam, and Mansour 2026); neither measures persistent residue, which AgentFootprint isolates.

A second line optimizes what agents keep *in context or in memory stores*. Memory systems (MemRefine (Kim et al. 2026)) compress memory banks for retrieval; context compaction (Slipstream (Chen et al. 2026)) shortens prompts. Both target the model-facing representation; the persistence layer can still checkpoint every intermediate underneath (one measured framework does). We measure the *system-wide residue*, coupled to context policy through α.

Trajectory platforms (LangSmith, Langfuse, AgentOps) persist agent runs as a product feature but publish no efficiency measures—potential reporters of such metrics, not sources. CLP (Rodrigues, Luo, and Yuan 2021) and zstd (Collet and Kucherawy 2021) compress syntactically (zstd is our baseline C); our semantic metrics capture what compression neither reveals nor repairs: what is duplicated, how retention scales, whether bytes reconstruct history.

Finally, the gap is explicit in the surveys: Mohammadi et al. (2025)'s metric catalog spans completion, step-efficiency, tool correctness, plan quality, latency, and cost—no storage dimension; GAIA, SWE-bench, and τ-bench report none. To our knowledge, AgentFootprint is the first systematic cross-framework benchmark of post-run storage footprint for LLM agents. Adjacent lines—workflow provenance (Davidson and Freire 2008; Herschel, Diestelkämper, and Ben Lahmar 2017), LSM storage amplification (O'Neil et al. 1996)—instrument different objects; we import their techniques as mitigations.

3 Storage Footprint Metrics

3.1 What Counts: Boundary and Taxonomy

We define the storage footprint of an agent execution—post-run local retention within the declared sandbox boundary—as all bytes present after the run, attributable to it, absent before it: files created or grown in the task workspace and every framework-side store—session/checkpoint databases, logs, snapshots, traces—including hidden home-directory state. *Footprint* and *retention* are used interchangeably; *residue* is the framework-side portion (all but agent-produced workspace artifacts). Environment artifacts shared across

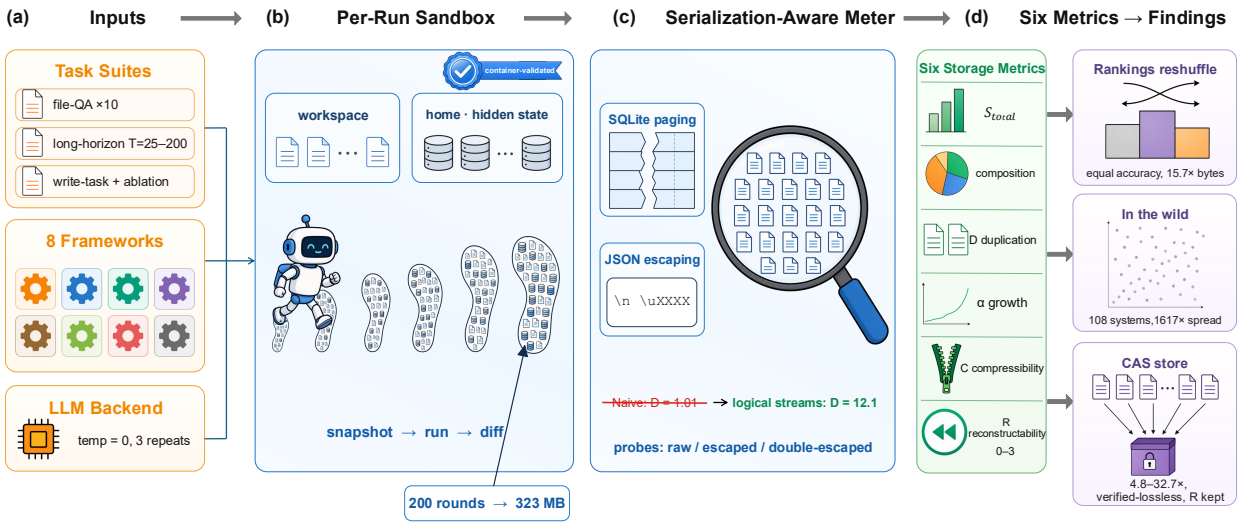

Figure 1: AGENTFOOTPRINT pipeline: each (framework, task, repetition) runs in a fresh sandbox; the filesystem delta is analyzed as *logical content streams* rather than raw bytes, yielding the six metrics of §3 that drive §5–§7.

tasks (model weights, tokenizer/package caches) are excluded, although their volume is reported for auditability. Measurement uses a per-run sandbox: a fresh workspace and home whose file inventory is snapshotted before and diffed after the run (§4.3); Supp. App. N audits this delta accounting over all 509 main-study sandboxes.

Retained bytes are not equally justified. We classify them as *decisions* (model outputs, generally not regenerable), *observations* (tool and environment returns, some of which are likewise unrecoverable), *derivations* (framework-generated intermediates: snapshots, summaries, checkpoints), and *duplicates* (the second through n-th appearance of the same bytes)—separating what must be kept, what may be referenced, and what is overhead. The taxonomy is conceptual; the benchmark reports deterministic storage-channel and duplication measurements, not automated semantic attribution of every byte.

3.2 Six Metrics

On this foundation we define six measurements per (framework, task) run:

S_{total} Retained bytes (apparent-length filesystem delta; Supp. App. N), divided into workspace artifacts and framework-side residue.

Store composition S_{total} grouped into state/checkpoints, debug/event logs, conversation/actions, and other channels by deterministic, adapter-audited path rules.

D Duplication factor $S_{\text{total}}^{\text{logical}} / S_{\text{unique}}$. Here, S_{unique} sums unique content-defined chunks (FastCDC (Xia et al. 2016) with SHA-256) over *logical streams*. Both live in logical content bytes (SQLite headers, indexes, and free pages enter neither); D=2 means the average content byte is stored twice; an exact-match lower bound under compressed or delta-encoded representations (Supp. App. K); physical-layout overhead is S_{total} vs. the logical total.

α Growth exponent from fitting $\log S_{\text{total}} \sim \alpha \log T$ at $T \in \{25, 50, 100, 200\}$ on the repeated-observation stress task—a descriptive statistic; per-horizon retention is primary (Supp. App. G).

C Compressibility: S_{total} divided by its size under zstd -19 with a 128 MB long-range window over the concatenated retained files (a compression bound, not an archive)—the syntactic "free lunch," reported so semantic metrics are read net of it.

R Conversation-history reconstructability, 0–3: given call boundaries (§5.4), can the retained bytes reconstruct the conversation history the model received at step k? R=0: nothing retained; R=1: bytes but no per-call structure; R=2: per-call structure but no complete copy of the observations; R=3: at least one channel holds a complete, exact record. Automated scoring yields a *candidate* grade; reported grades additionally require the adapter's serialization contract to pass the field-by-field reconstruction of §5.4 (construct check: Supp. App. O).

3.3 Measurement Methodology: Serialization Masks Duplication

A key methodological finding is that raw-file chunking substantially understates duplication. Running content-defined chunking over raw retained files reports $D = 1.01$ for LangGraph's checkpoint store—yet zstd compresses that store 142×, a contradiction. Two serialization mechanisms mask the duplication. SQLite fragments large values across 4 KB pages interleaved with headers and pointers, so chunk boundaries never align across copies; and JSON escaping rewrites the bytes of embedded content (`\n`, `\uXXXX`), so trajectory copies no longer match the source bytes. Our meter therefore extracts *logical streams* before fingerprinting: every SQLite cell and JSONL line is chunked as its own stream, while other files are treated as whole streams. Under logical streams the same store measures $D = 12.1$ (a single store; Table 1's 30-

run means are D 12.8, C 136)—the duplication was present but obscured by serialization, which may partly explain why this axis has previously gone unmeasured.

Cross-representation copies still evade chunk matching, so we add *content probes*: fixed substrings and whole lines sampled deterministically from every input file, searched in all logical streams under raw and JSON-escaped encodings; doubly-nested encodings are only partially detected (Supp. App. K). The mean occurrence count (*echo*) estimates how many times the input content is stored; the same probes power the R score's completeness check, and whole-line probes survive re-formatting such as line-numbered tool outputs. Supp. App. K calibrates D and echo on synthetic stores with known duplication.

4 The AgentFootprint Benchmark

4.1 Task Suites

AgentFootprint uses six task families, each isolating one storage behavior, plus a control. The *file-QA suite* (10 tasks) gives the agent ten ∼60 KB documents (600 KB total) and five sequential questions whose answers are unique registry codes embedded in the files; questions revisit files in an A,B,A,C,A pattern with instructed re-reading (read amplification). Grading is exact substring match—no LLM judge. The *long-horizon suite* fixes a 2 KB status file re-read for $T \in \{25, 50, 100, 200\}$ rounds, isolating α from content growth. The *write-task suite* (5 tasks) reads two documents and produces a summary file via a `write_file` tool (legitimate artifacts vs. residue). The *exploratory-retrieval suite* (5 tasks) removes every adversarial element (§5.5). Two *heterogeneous families* (five tasks each)—*edit* and *data analysis*—are read–transform–write tasks graded *jointly*: correct answer *and* artifact delivered in the shared workspace (§5.5). Finally, *ablation runs* re-execute the file-QA suite with persistence disabled.

4.2 Frameworks and Fairness Protocol

We measure eight representative frameworks: LangGraph 1.2.8 (LangChain AI 2026), AutoGen 0.7.5 (Microsoft 2025), CrewAI 1.15.2 (CrewAI Inc. 2026), SmolAgents 1.26.0 (Hugging Face 2026), OpenAI Agents SDK 0.18.0 (OpenAI 2026), LlamaIndex 0.14.23 (LlamaIndex Inc. 2026), Agno 2.7.1 (Agno AGI 2026), and InfiAgent 3.12.24 (Yu et al. 2026). The roster covers widely used open-source agent APIs with distinct persistence mechanisms; Agno and InfiAgent provide two independent implementations of windowed context management (a runtime forgetting loop; a bounded per-request window), so the growth study (§5.2) can compare full-history and bounded-history policies.

Fairness comes from configuring each framework in its *documented durable-session configuration*—the persistence setup its own documentation describes, selected by a predefined protocol—not from tuning any of them for storage. Table 1 lists the measured mechanisms. All frameworks receive identical task text, the same minimal tools (list, read, write where applicable), the same backend model (DeepSeek-V4-Flash, served by one provider, temperature 0), and three independent repetitions. We measure the storage produced by the documented default configuration; the exact adapter code ships in the artifact, and §8 reports a configuration-sensitivity check (AutoGen's save cadence).

This protocol measures the *system-level default footprint*—what the documented configuration leaves on disk. Its limit: these documented configurations support different capabilities (Supp. App. R), so equal-R comparisons rank what defaults retain for the same reconstructability score, not capability-normalized efficiency (§8).

4.3 Harness

The harness (Figure 1) runs each (framework, task, repetition) in a fresh sandbox: the workspace holds the task corpus; `HOME`/XDG paths are redirected into the sandbox (hidden state captured); the adapter runs the framework's quickstart-style agent; the meter diffs both inventories. Re-running LangGraph (three tasks) in per-run Docker containers changed measurements by 0–8.5%, below the 11% repetition-level standard deviation. Harness, generators, adapters, and per-run records ship in the artifact; Supp. App. C enumerates all 1,061 runs and states the statistical units and aggregation conventions.

5 Controlled Results

5.1 Same Accuracy, $15.7\times$ the Bytes

Table 1(a) presents the file-QA suite (eight frameworks, ten tasks, three repetitions): among the five persisting configurations with 100% task accuracy, documented defaults span $15.7\times$ in retained bytes (bootstrap 95% CI $[15.0, 16.3]$; Supp. App. C). Across the six R=3 systems, minimal history-retaining forms span $3.9\times$—three executed variants, two native forms, and one disclosed post-hoc split (§5.4; Supp. App. Q). Across the full roster, Agno answers 99.3% and AutoGen 86%; the other six answer every question correctly. Persisting frameworks retain from 0.32 MB (CrewAI) to 5.10 MB (LangGraph) per task on a 600 KB corpus, and SmolAgents retains exactly zero bytes. Composition (Table 1b) shows retention concentrated in different channels—checkpoints (LangGraph), event logs (AutoGen), conversation stores and debug traces (InfiAgent; setup staging excluded—including it raises its mean to 2.72 MB, above AutoGen; Supp. App. N). The duplication factor separates architectural families: full-history checkpoint/snapshot designs (D: LangGraph 12.8, AutoGen 8.4, LlamaIndex 4.2) are most redundant, while windowed or append-only designs stay below 2.6. The echo metric makes it concrete: probes from the same files, read equally often, appear $8.1\times$ under LangGraph but only $0.5\times$ under OpenAI Agents.

5.2 Growth Regimes under Repeated Observations

On the controlled repeated-observation stress task, the retention curves fall into three regimes (Figure 2; α in Table 1; three seeded runs per horizon for the full-history trio). Frameworks that persist the full message history at every step—LlamaIndex, AutoGen, LangGraph (α 1.74–1.95)—grow superlinearly: monitoring one 2 KB status file

Table 1: Main results: the eight measured configurations as one profile. **(a)** File-QA suite (10 tasks × 3 repetitions; S_{total} mean±SD over 30 runs; decomposition in Supp. App. C; spreads on unrounded values). Mechanism: documented durable-session persistence (§4.2); α: growth exponent on the stress task (mean±SD over three seeds for the full-history trio, single run otherwise; §5.2, Supp. App. G); R: modal reconstructability score, reconstruction-validated (§5.4); C: zstd long-range compressibility; CAS: post-CAS retention (§7), first-repetition subset (factor in parentheses). **(b)** Channel composition of S_{total}, process-boundary resume, and full-roster backend replication: per-task S_{total} under two further backends, rank correlation with the main study ρ=0.96 / 0.89 (Supp. Apps. B, F, J).

Framework	Persistence mechanism	S_{total} ↓	D ↓	Echo ↓	C	α	Acc. ↑	R ↑	CAS ↓
LangGraph	SQLite checkpointer	5.10±0.58 MB	12.8	8.1×	136	1.74±0.01	100%	3	0.50 MB (10.3×)
AutoGen	per-turn state + event log	2.49±0.33 MB	8.4	4.1×	106	1.95±0.18	86%	3	0.07 MB (32.7×)
InfiAgent	task stores + raw traces	2.12±0.05 MB	1.8	1.7×	43	1.15	100%	3	0.36 MB (5.8×)
Agno	SQLite session db	1.15±0.02 MB	2.5	0.8×	42	0.98	99.3%	3	0.06 MB (20.5×)
LlamaIndex	serialized `Context`	0.93±<0.01 MB	4.2	1.3×	40	1.88±0.10	100%	3	0.09 MB (10.3×)
OpenAI Agents	`SQLiteSession`	0.33±0.02 MB	1.6	0.5×	14	0.73	100%	3	0.07 MB (4.8×)
CrewAI	task-output store	0.32±<0.01 MB	1.5	0.5×	14	1.20	100%	1	0.05 MB (6.1×)
SmolAgents	none (in-memory)	0 B	–	–	–	–	100%	0	–

(b) Channel (MB/run)	LangGraph	AutoGen	InfiAgent	Agno	LlamaIndex	OpenAI Ag.	CrewAI	SmolAgents
State/checkpoints	5.10	0.83	0.07	1.15	0.93	0.33	0.32	–
Debug/event logs	–	1.65	0.86	–	–	–	–	–
Conversation/actions	–	–	1.20	–	–	–	–	–
Resume in fresh process	✓	✓	✓	✓	✓	✓	×	–
S_{total}, Qwen3.6-27B (MB)	5.20	2.64	2.13	1.16	0.94	0.31	0.32	0
S_{total}, Qwen2.5-7B (MB)	2.49	1.26	2.14	0.77	1.20	0.21	0.37	0

for 200 rounds leaves 323±26 MB under LangGraph (D ≈49), 102±21 MB under AutoGen. Frameworks with windowed context (InfiAgent's forgetting loop) or bounded per-round stores (CrewAI, Agno) stay near-linear (α 0.98–1.20): at T=200 (completed by every persisting framework) they retain 1.4–36.4 MB against LangGraph's 323±26 MB.

Accuracy stays high throughout (median 195 of 200 at T=200 across persisting frameworks and seeds; minimum 171)—growth differences are not task-failure artifacts—and α is a workload-conditioned full-range fit: on seed 1, local slopes over T=25→100 are 1.96–1.98 for all three heavy frameworks, but over 100→200 they diverge (LlamaIndex 1.92, AutoGen 1.38, LangGraph 0.96) with no failed rounds and token volumes far below context limits (per-seed spreads in Supp. App. G). Checkpoint-layer consolidation is the suspected cause, and the repetitions show that variability concentrates in absolute volume (LangGraph T=100: 153±10 MB) while the fitted exponent remains stable (1.74±0.01). Four horizons at three runs each remain insufficient to distinguish among candidate functional forms (Supp. App. G); the claim is superlinear growth over the measured range—every seed of every full-history framework fits $\alpha \geq 1.72$—not a scaling law. Archival policy matters as much as architecture: LlamaIndex's newest-snapshot-only form leaves 85–645 KB ($\alpha \approx 0.98$)—superlinear growth follows from the *append-all* archival policy, not from durable resumption (Supp. App. G).

On this fixed-content workload, read amplification and storage amplification are separable, and the exponent tells them apart. OpenAI Agents is sublinear (α=0.73): its session log appends each message once rather than re-serializing the whole history, so the status file's content accumulates linearly with reads (echo=120 at T=200) with no redundancy beyond that (D=1.0). On this workload the exponent therefore reflects context-retention policy: whether a framework windows its history is observable in how its disk usage scales. We do not claim the fitted exponent transfers to arbitrary workloads.

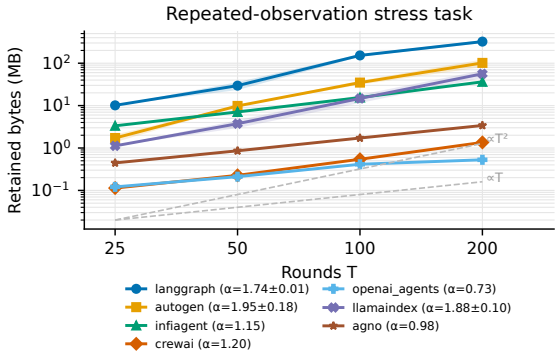

Figure 2: Retention growth on the repeated-observation stress task (log–log). Lines: means; bands: ±1 SD over three seeds (full-history trio); α: per-seed full-range fit, mean ± SD. Windowed and append-only designs remain linear or sublinear.

5.3 Persistence Ablations

Retention is not required for immediate task success on this short retrieval suite: with persistence disabled, five frame-

works (LangGraph, AutoGen, LlamaIndex, OpenAI Agents, Agno; five-task subset) keep their accuracy—100% for four, AutoGen within run variance (22/25 vs. 19/25)—and retain exactly zero bytes. Retention supports replay, resumption, and recovery rather than immediate task completion, and the ablated configurations—like SmolAgents' default—provide none ($R{=}0$). InfiAgent exposes no debug-channel switch, so we perform a post-hoc ablation by excluding its debug channel, which accounts for 40% of its footprint; removal *raises* D 1.75→2.1. The ablation also sharpens what the metrics measure: disabling persistence drives S_{total} to zero and R to 0 together, so the metric pair distinguishes a configuration change from a structural property.

5.4 Reconstructability

Scoring one run per (framework, task) pair (80 runs), six frameworks achieve $R{=}3$ at retained sizes spanning $15\times$; CrewAI retains bytes without per-call structure ($R{=}1$), SmolAgents nothing. InfiAgent's raw-trace channel supplies what its truncated debug log lacks—multi-channel designs need channel-level scoring. We validate the grade by *full conversation-history reconstruction*: each $R{=}3$ framework re-ran three tasks behind a logging proxy recording every request body (194 calls); we rebuilt the conversation from the retained bytes and compared field-by-field (roles, contents, tool-call ids/arguments, results, order): with call boundaries taken from the proxy log, 194/194 calls are exactly reconstructible as contiguous subsequences (contracts in Supp. App. I). System prompts are recoverable from retention in Agno and InfiAgent; the rest keep them in code-side configuration (LangGraph configures none). A *process-boundary resume probe* (Supp. App. J) confirms the grades operationally: every $R{=}3$ store resumes in a fresh process; CrewAI's ($R{=}1$) does not.

Supp. App. S plots default and post-CAS retention with R and accuracy; two observations follow. First, the same reconstructability score—$R{=}3$—is obtained at retained sizes from 0.33 MB (OpenAI Agents) to 5.10 MB (LangGraph), and content addressing reduces all seven persisting configurations to 0.05–0.50 MB (0.06–0.50 MB among the $R{=}3$ configurations): most of the spread is removable redundancy, not a requirement of the score. R scores history reconstruction, not every capability a store may serve (crash resume, workflow state, debug provenance), so the spread is unexplained by reconstructability alone, not established as pure waste (§8); the executed minimal history-retaining forms (§5.1) converge at 0.31–0.33 MB with accuracy and R unchanged (Supp. App. Q). Second, accuracy alone does not expose these storage differences: six frameworks score 100%, and Agno scores 99.3%. A success-only leaderboard penalizes AutoGen for its lower accuracy (86%) but not for its 2.49 MB footprint, and it treats LangGraph as fully successful despite retaining $15.7\times$ CrewAI's bytes. Section 6 examines the storage–success relationship at scale.

5.5 Beyond Adversarial Retrieval: Exploratory and Heterogeneous Tasks

The stress suites deliberately provoke re-reads, so an *unprompted exploratory* check removes every adversarial element: five multi-document tasks, free exploration via `list_files`. Dispersion persists (23/28 pairwise orderings agree; persisting frameworks span $8.1\times$) and echo *rises* (LangGraph $8.1\rightarrow11.8\times$)—amplification is not an artifact of adversarial design (AutoGen 47% and SmolAgents 67% differ in accuracy and are excluded from equal-accuracy comparisons; Supp. App. E).

The two heterogeneous families (§4.1; all eight frameworks, 80 runs; Supp. App. D) grade success *jointly*: the answer must be correct *and* the required artifact delivered in the shared workspace, validated exactly against the task specification. Three patterns from the controlled suites reappear. Equal-success dispersion: five configurations achieve full joint success on both families—LangGraph, CrewAI, OpenAI Agents, LlamaIndex, Agno—and span $6.9\times$ in retention on edit tasks (24–164 KB) and $14.1\times$ on data tasks (8.8–124 KB); answer-only accuracy is broader (seven frameworks reach 100% on edit) but conflates delivered and undelivered work. The mechanism: LangGraph's checkpointer echoes the input CSVs $7.8\times$ on data tasks—observation amplification is not a file-QA artifact. The evaluation boundary: InfiAgent answers every question correctly yet fails joint success on all ten tasks, because it deposits the deliverable in its managed home-side store rather than the shared workspace—exactly what the workspace/home split of §3 makes visible; and AutoGen's default tool-iteration budget again fails the read–transform–write chain (0% on both families).

5.6 Fixed-Trace Control: Isolating the Persistence Layer

All preceding suites entangle agent policy with storage encoding. A scripted mock endpoint (in the artifact) replays one identical trajectory (three reads, 182.7 KB of observations, one answer) through the seven persisting frameworks' unmodified adapters: the content is fixed, so the framework configuration is the only variable. Identical-content amplification spans $1.03\times$ (LlamaIndex) to $6.86\times$ (InfiAgent) under the batched transport protocol (sequential transport raises LangGraph to $7.40\times$; Supp. App. L)—a $6.7\times$ spread with no contribution from agent behavior; what remains is the configuration-level persistence mechanism—which objects are stored, how often, in what serialization. The protocol is deterministic, offline, and model-free (Supp. Apps. L–M).

6 Storage in the Wild

Controlled suites could exaggerate differences that vanish on real workloads, so we compare with exported trajectories from SWE-bench Verified (Jimenez et al. 2024). We harvested all 134 submissions in our July-2026 snapshot; keys are grouped by canonical instance ID (S3 layouts vary). Of 115 submissions with S3 prefixes, 113 expose size listings and 108 meet the 90%-coverage inclusion thresholds (instances and mapped bytes); 19 lack a public prefix, seven cannot be instance-normalized. Each system's trajectory volume is a *per-system lower bound* on that system's footprint—it is the exported artifact, not the on-disk residue. Ratios between systems inherit no such guarantee (export policies differ): the spread corroborates heterogeneity, not runtime footprints.

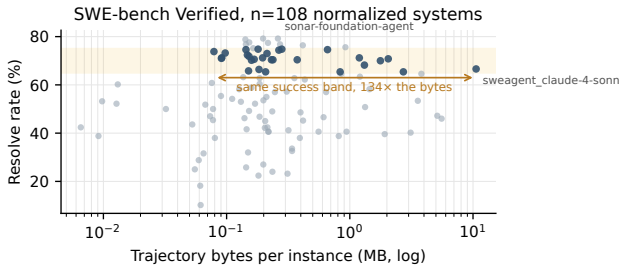

Figure 3: SWE-bench Verified, 108 instance-normalized systems: resolve rate vs. exported trajectory bytes per instance (log). No detectable volume–success correlation (τ_b=0.080, p=0.222).

Mean exported trajectory volume per mapped instance ranges from 6.6 KB to 10.6 MB across systems—1,617× (Figure 3). In the direct test, resolve rate shows no detectable correlation with bytes per instance (Kendall τ_b=0.080, permutation p=0.222): within this population, exporting more trajectory data is not associated with resolving more issues. The 28 systems in the 65–75% band alone span 134× (IQR 139–402 KB; the spread is not driven solely by extremes). Because the 108 submissions include multiple versions per team or family (63 families; 25 multi-entry), we repeat the direct test under two analyses that respect family-level dependence (latest-per-family and family means; $\tau_b \leq 0.07$, $p \geq 0.40$)—the conclusion is unchanged. Detailed inspection of six sampled submissions shows intra-trajectory duplication from 1.0 (deduplicated exports) to 3.35.

7 A Reference Mitigation: Content-Addressed Trajectory Store

Because duplication drives much of the footprint, we evaluate content addressing—store each block once, by hash—as a feasibility reference, not a novel compression algorithm. Our implementation (about 300 lines, framework-agnostic) post-processes a run's retained stores: strings and blobs above 1 KB inside JSON, JSONL, log lines, and SQLite cells are moved into a zstd-compressed content-addressed store and each is replaced by a hash reference with preview; rehydration inverts the transform on demand. The design preserves each framework's store schema—composing with, not replacing, existing persistence layers.

Across 70 runs of the seven persisting frameworks, the compactor reduces retention by 4.8–32.7× (Table 1, CAS column), and all 310 retained files pass the restore checks (JSON-object equality for structured text; row-set equality for SQLite user tables, internal/FTS shadow tables excluded as regenerable; pass-through byte-exact). Re-scoring R on the CAS-restored stores reproduces every framework's original score (R=3 for all six, R=1 for CrewAI)—we verify preservation rather than assume it. The compactor achieves its largest reduction on AutoGen (32.7×): its event log and per-turn snapshots repeat the same messages—exactly what content addressing removes.

Relation to whole-store compression. A zstd archive is smaller than the CAS representation (LangGraph: ≈0.04 vs. 0.50 MB), but answers a different question: C measures an *opaque archive*, unusable until fully decompressed, while the CAS output remains a *schema-preserving* store—blocks stay hash-addressable and can be restored on demand (consuming references in place needs a framework-side shim we do not build; R is re-scored on restored stores); the two compose (Supp. App. H). Garbage collection, deletion semantics, and cross-tenant privacy are out of scope by design.

The write-task suite quantifies how far defaults are from this floor: a 39-byte summary file (a deliberately small write task) leaves 128 KB–1 MB of residue, up to 24,033× the artifact (the production case of §1, with realistic outputs, shows ∼30×). With the ablations, these bracket the design space: retaining nothing yields no reconstructability, while defaults can retain megabytes even though the same score is achievable within 50–500 KB after post-processing.

8 Discussion and Limitations

Isolation and backend transfer. Containerized and sandbox measurements differ by 0–8.5%. Full-roster replications preserve the storage ranking on Qwen3.6-27B (ρ=0.96; sizes 0.93–1.06×) and weak Qwen2.5-7B (ρ=0.89 despite accuracy collapse), indicating that persistence design dominates backend choice (Table 1b; Supp. App. F).

Scope. Six audited boundaries apply: (i) controlled suites cover retrieval and retention, not general reasoning; (ii) results describe versioned configurations, not frameworks in the abstract; (iii) wild data are exports, not runtime residue; (iv) R covers history only; (v) the roster is representative and α workload-conditioned; (vi) fresh sandboxes *understate* shared threads (95.2 vs. 5.8 MB; Supp. App. M).

Configuration sensitivity. AutoGen's save cadence is the most contestable choice; an end-of-run variant remains similar (2.1 vs. 2.49 MB) and superlinear (α 1.50).

S_{total} and R must be read jointly: either can be gamed by retaining nothing or everything; the target is exact reconstruction at minimal bytes (§7).

Why bytes matter. The concern is less disk cost than retention obligations: at 10^4 tasks/day, measured defaults produce 51 versus 3.1 GB/day for the executed history-equivalent form (Supp. App. Q), a 16× gap at equal R; deletion forfeits supported capabilities (§5.3).

Ethics. Corpora are synthetic; wild data are public; CAS obligations appear in Supp. App. A.

9 Conclusion

Agent evaluations report task success, reliability, and inference cost, but not persistent storage. AgentFootprint makes this axis reportable through a serialization-aware suite, controlled and in-the-wild studies, and a reconstruction-preserving store, exposing the 15.7× equal-accuracy and 6.7× fixed-trace spreads alongside inference cost.

AI-use disclosure. Claude (Anthropic) assisted with parts of the experiment code and with manuscript polishing; the authors verified and take responsibility for all content.

References

Agno AGI. 2026. Agno. https://pypi.org/project/agno/2.7.1/. Version 2.7.1; accessed: 2026-07-13.

Chen, Z.; Pan, R.; Dai, Y.; and Netravali, R. 2026. Slipstream: Trajectory-Grounded Compaction Validation for Long-Horizon Agents. arXiv:2605.08580.

Collet, Y.; and Kucherawy, M. 2021. Zstandard Compression and the 'application/zstd' Media Type. RFC 8878, RFC Editor.

CrewAI Inc. 2026. CrewAI. https://pypi.org/project/crewai/1.15.2/. Version 1.15.2; accessed: 2026-07-13.

Davidson, S. B.; and Freire, J. 2008. Provenance and Scientific Workflows: Challenges and Opportunities. In *Proceedings of the ACM SIGMOD International Conference on Management of Data*, 1345–1350. ACM.

Herschel, M.; Diestelkämper, R.; and Ben Lahmar, H. 2017. A Survey on Provenance: What for? What Form? What from? *The VLDB Journal*, 26(6): 881–906.

Hugging Face. 2026. smolagents. https://pypi.org/project/smolagents/1.26.0/. Version 1.26.0; accessed: 2026-07-13.

Jimenez, C. E.; Yang, J.; Wettig, A.; Yao, S.; Pei, K.; Press, O.; and Narasimhan, K. R. 2024. SWE-bench: Can Language Models Resolve Real-World GitHub Issues? In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Kapoor, S.; Stroebl, B.; Siegel, Z. S.; Nadgir, N.; and Narayanan, A. 2025. AI Agents That Matter. *Transactions on Machine Learning Research*.

Kim, M.; Baek, J.; Jeong, S.; and Hwang, S. J. 2026. MemRefine: LLM-Guided Compression for Long-Term Agent Memory. arXiv:2606.13177.

LangChain AI. 2026. LangGraph. https://pypi.org/project/langgraph/1.2.8/. Version 1.2.8; accessed: 2026-07-13.

LlamaIndex Inc. 2026. LlamaIndex. https://pypi.org/project/llama-index-core/0.14.23/. Core version 0.14.23; accessed: 2026-07-13.

Mialon, G.; Fourrier, C.; Swift, C.; Wolf, T.; LeCun, Y.; and Scialom, T. 2024. GAIA: A Benchmark for General AI Assistants. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Microsoft. 2025. AutoGen. https://pypi.org/project/autogen-agentchat/0.7.5/. Version 0.7.5; accessed: 2026-07-13.

Mohammadi, M.; Li, Y.; Lo, J.; and Yip, W. 2025. Evaluation and Benchmarking of LLM Agents: A Survey. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V.2*, 6129–6139. ACM.

O'Neil, P.; Cheng, E.; Gawlick, D.; and O'Neil, E. 1996. The Log-Structured Merge-Tree (LSM-Tree). *Acta Informatica*, 33(4): 351–385.

OpenAI. 2026. OpenAI Agents SDK. https://pypi.org/project/openai-agents/0.18.0/. Version 0.18.0; accessed: 2026-07-13.

Orogat, A.; Rostam, A.; and Mansour, E. 2026. Understanding Multi-Agent LLM Frameworks: A Unified Benchmark and Experimental Analysis. arXiv:2602.03128.

Rodrigues, K.; Luo, Y.; and Yuan, D. 2021. CLP: Efficient and Scalable Search on Compressed Text Logs. In *15th USENIX Symposium on Operating Systems Design and Implementation (OSDI 21)*, 183–198. USENIX Association. ISBN 978-1-939133-22-9.

Wang, N.; Hu, X.; Liu, P.; Zhu, H.; Hou, Y.; Huang, H.; Zhang, S.; Yang, J.; Liu, J.; Zhang, G.; Zhang, C.; Wang, J.; Jiang, Y. E.; and Zhou, W. 2025. Efficient Agents: Building Effective Agents While Reducing Cost. arXiv:2508.02694.

Xia, W.; Zhou, Y.; Jiang, H.; Feng, D.; Hua, Y.; Hu, Y.; Zhang, Y.; and Liu, Q. 2016. FastCDC: A Fast and Efficient Content-Defined Chunking Approach for Data Deduplication. In *2016 USENIX Annual Technical Conference (USENIX ATC 16)*, 101–114. Denver, CO: USENIX Association. ISBN 978-1-931971-30-0.

Yao, S.; Shinn, N.; Razavi, P.; and Narasimhan, K. R. 2025. τ-bench: A Benchmark for Tool-Agent-User Interaction in Real-World Domains. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Yu, C.; Wang, Y.; Wang, S.; Yang, H.; and Ming, L. 2026. InfiAgent: An Infinite-Horizon Framework for General-Purpose Autonomous Agents. In *Findings of the Association for Computational Linguistics: ACL 2026*, 35884–35894. San Diego, California, United States: Association for Computational Linguistics.

Supplementary Material for "The Hidden Footprint: Making Storage a First-Class Metric for LLM Agent Evaluation"

Chenglin Yu[1], **Hongquan Gui**[1], **Ying Yu**[2], **Hongxia Yang**[3], **Ming Li**[1,4*]
[1]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University
[2]College of Economics and Management, Zhejiang Normal University
[3]Department of Computing, The Hong Kong Polytechnic University
[4]Research Institute for Generative AI, The Hong Kong Polytechnic University
chenglin.yu@polyu.edu.hk, ming.li@polyu.edu.hk

This document contains Appendices A–S accompanying the main paper.

A Availability and Artifact Contents

The complete benchmark code—harness, task generators, all eight framework adapters, the CAS reference implementation, and the analysis scripts—is included in the anonymized code-and-data supplement accompanying this submission under the MIT license. A revision-matched supplementary artifact accompanying this submission contains the synthetic task corpora, per-run meter summaries, replay/CAS outputs, horizon and write-task measurements, Docker cross-check, instance-normalized Tier-B cache, inclusion diagnostics, and figure/statistics scripts. Material exceeding the appendix (per-run measurement records, reconstruction evidence, and raw suite outputs) ships in that supplementary archive rather than in the PDF, together with a self-contained README, pinned analysis dependencies, a version-alignment manifest, and one complete retained store per persisting framework (byte-exact for six; InfiAgent's sanitized of key-material placeholders and operator path strings by length-preserving substitution, with re-measured equivalents) so that D, C, composition, and CAS restore can be re-measured independently. The in-the-wild data (§6 of the main paper) was collected from the SWE-bench experiments repository and its public submission buckets: we downloaded only submission metadata, published results files, and object-size listings, plus sampled trajectories for six submissions—all artifacts that the leaderboard's submission policy already requires to be public—and we release the instance-normalizing harvest script, cached size tables, inclusion diagnostics, and statistical analysis for exact reproduction. Synthetic corpora and generators carry the repository's MIT license and contain no personal data; SWE-bench-derived size tables inherit the leaderboard's public-data terms. The submitted benchmark artifact is revision-matched to this manuscript: framework adapters pin exact package versions with per-framework lockfiles, results are grouped by benchmark and backend-model version, and new framework versions enter as new result rows rather than overwriting prior ones.

Continuous verification. The repository runs an automated verification suite on every commit: the meter calibration (Appendix K) and the R construct check (Appendix O) re-execute from scratch, the main file-QA table and the heterogeneous joint-success cells are regenerated from the archived per-run records and compared against the published values, and the fixed-trace protocol (Appendix L) runs end-to-end against the mock endpoint—an offline conformance path requiring no model access. A container image built from the lockfiles covers seven of the frameworks; InfiAgent installs from PyPI, and `envlocks/` records the study versions.

CAS deployment obligations. The reference compactor does not resolve the obligations a production content-addressed store inherits: unsalted content hashes can leak content equality across trajectories or tenants; shared blobs complicate deletion requests (deletion must garbage-collect unreferenced blobs, not merely drop references); and deduplication across trust boundaries can enable existence probes against low-entropy content. Deployments should scope stores per tenant, salt hashes where equality itself is sensitive, and couple deletion with reference-counted collection.

B Per-Framework Composition and Persistence Configuration

Table 1 gives the numeric decomposition behind Figure 1, together with each framework's persistence mechanism as configured in the harness (the exact adapter code ships in the artifact).

*Corresponding author.

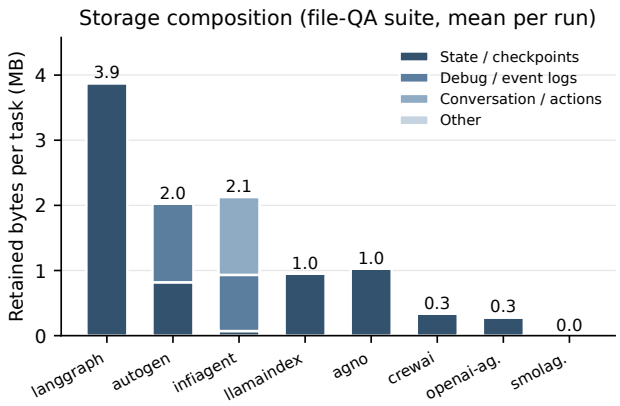

Figure 1: Composition of retained bytes on the file-QA suite. Frameworks are heavy in different places: checkpoints (LangGraph), debug/event logs (AutoGen), conversation stores (InfiAgent).

Table 1: Retained MB per file-QA run by store category (mean over 30 runs), and the persistence mechanism measured.

Framework	State	Dbg/log	Conv.	Mechanism
LangGraph	5.10	–	–	SQLite checkpointer
AutoGen	0.83	1.65	–	per-turn state, event log
InfiAgent	0.07	0.86	1.20	task stores, raw traces
Agno	1.15	–	–	SQLite session db
LlamaIndex	0.93	–	–	serialized `Context`
OpenAI Agents	0.33	–	–	`SQLiteSession`
CrewAI	0.32	–	–	task-output store
SmolAgents	–	–	–	none (in-memory)

Channel asymmetries. AutoGen's documented durable-session recipe pairs per-turn `save_state` with the tutorial's event logger; the channels are separable above: without the event log its file-QA retention is 0.83 MB (below Agno), and the growth class is unchanged—both channels are superlinear on the stress task (event log α=1.70, snapshot series α=1.86; the log records each turn's full request). OpenAI Agents' hosted tracing is disabled because it is *remote* retention—outside the local boundary of §3.1 of the main paper by definition—and would upload task content to a third-party service.

Write-task residue by framework. Mean residue over the five write tasks: LangGraph 920 KB, InfiAgent 1,022 KB, AutoGen 374 KB, Agno 285 KB, OpenAI Agents 149 KB, CrewAI 136 KB, LlamaIndex 128 KB, SmolAgents 0 B; per-run records ship in the artifact.

C Run Accounting

Table 2 enumerates every sandboxed run behind the reported numbers: a 509-run main study plus 552 replication and variant runs—the two full-roster backend replications (Appendix F), the growth repetition seeds (Appendix G), the executed history-only forms (Appendix Q), the continuous shared-store threads (Appendix M; long-lived by design, measured cumulatively), and the two later-added proxied tasks per R=3 framework (Appendix I; the original single-task round is in the main-study row)—for a grand total of 1,061 sandboxes. Each fresh-sandbox run carries its own baseline snapshot and measurement record, and all records ship in the supplementary archive.

Computing infrastructure. All controlled runs executed on one Apple M5 workstation (16 GB RAM, macOS 26.4, arm64; Python 3.12), one framework process at a time; each per-run record stores the platform and Python version, and `envlocks/` pins framework and library versions. The reported storage metrics are byte counts and do not depend on hardware.

Statistical units and aggregation. The framework–task pair is the primary comparison unit; the three repetitions quantify run-level variability. Recorded model calls (§5.4 of the main paper) and retained files (§7 of the main paper) are verification objects, not independent samples, and the 108 in-the-wild submissions carry family-level dependence, which §6 of the main paper's sensitivity analyses address. Ratio metrics (D, echo, C) are computed per run and then averaged—mean of per-run ratios, not ratio of pooled sums—so table entries are per-run expectations. Decomposing main-paper Table 1's dispersion, between-task SD of task means versus mean within-task repetition SD is, in MB: LangGraph 0.27/0.36, AutoGen 0.16/0.25, InfiAgent 0.03/0.02, Agno 0.01/0.01, LlamaIndex 0.00/0.00, OpenAI Agents 0.01/0.01, CrewAI 0.00/0.00—repetition variability dominates for the two heaviest frameworks; task-content variability is second-order elsewhere. The in-the-wild study adds no sandboxed runs; it harvests public metadata for 134 submissions, of which 108 pass instance normalization. SmolAgents retains zero bytes at every measured horizon; its T=200 point was not run. A nonparametric bootstrap (20,000 resamples of run means within each framework) puts a 95% CI of $[15.0, 16.3]$ on the headline 15.7× file-QA spread. The executed history-only 3.9× is a ratio of 10-task means (per-configuration SDs in Appendix Q); the fixed-trace 6.7× comes from a deterministic single-trajectory protocol and is reported as a point estimate.

Table 2: Complete run accounting. No run is shared between rows.

Experiment	Runs	Composition
File-QA main study	240	8 fw × 10 tasks × 3 reps
Persistence ablations	35	6 fw × 5; InfiAgent 2 variants
AutoGen end-of-run save	8	5 file-QA + 3 horizon points
Second backend (4o-mini)	18	3 fw × 5 QA + 3 at T=50
Replay-verification source	6	1 proxied task per R=3 fw
Long-horizon	31	7 fw × 4 T + SmolAgents × 3
Write tasks	40	8 fw × 5 tasks
Exploratory retrieval	40	8 fw × 5 tasks
Edit family	40	8 fw × 5 tasks
Data-analysis family	40	8 fw × 5 tasks
Docker cross-check	3	LangGraph × 3 tasks
Fixed-trace (batched)	7	7 fw × 1 scripted trace
Fixed-trace (sequential)	1	LangGraph transport variant
Main-study subtotal	509	
Backend: Qwen3.6-27B	240	8 fw × 10 tasks × 3 reps
Backend: Qwen2.5-7B	240	8 fw × 10 tasks × 3 reps
Growth repetition seeds	24	3 fw × 4 T × 2 seeds
Executed history-only	30	3 configurations × 10 tasks
Continuous shared threads	6	6 fw × ten-task thread
Replay-source additions	12	2 further proxied tasks × 6 fw
Grand total	1,061	

D Heterogeneous Task Families: Per-Family Results

Tables 3 and 4 give the per-framework results behind §5.5 of the main paper (one run per task, five tasks per family; results are descriptive and no repetition variance is reported; S_{total} is the mean over the five tasks and includes the delivered artifact itself, under 1 KB in every case). "Acc" is answer accuracy; "Artifact" counts tasks whose required artifact passes a *spec-exact* validator in the shared workspace (edit: every DATE line ISO and set-equal to the source ledger; data: exact header, all product totals correct, descending order); "Joint" counts tasks where both hold—the task-success criterion of §5.5 of the main paper. An earlier presence-based checker over-credited SmolAgents' data artifacts (3/5); spec-exact grading yields 1/5, and no other cell changes. InfiAgent's home-side artifacts pass the spec-exact check as well.

Table 3: Edit family (normalize ledger dates, deliver corrected copy). [a]InfiAgent's artifacts are all content-correct but deposited in its managed home-side store rather than the workspace. [b]AutoGen exhausts its default tool-iteration budget before the write step.

Framework	Acc	S_{total} (KB)	D	Echo	Artifact	Joint
LangGraph	100%	164	1.6	4.8×	5/5	5/5
AutoGen[b]	0%	6.1	1.0	0.0×	0/5	0/5
InfiAgent[a]	100%	287	1.2	2.9×	0/5	0/5
CrewAI	100%	41	1.0	1.2×	5/5	5/5
SmolAgents	100%	2.9	1.0	0.2×	4/5	4/5
OpenAI Agents	100%	44	1.0	0.8×	5/5	5/5
LlamaIndex	100%	24	1.0	0.9×	5/5	5/5
Agno	100%	66	1.0	1.4×	5/5	5/5

Table 4: Data-analysis family (aggregate two sales CSVs, deliver `summary.csv`). Footnotes as in Table 3.

Framework	Acc	S_{total} (KB)	D	Echo	Artifact	Joint
LangGraph	100%	124	1.4	7.8×	5/5	5/5
AutoGen[b]	0%	6.4	1.0	0.0×	0/5	0/5
InfiAgent[a]	100%	284	1.2	0.0×	0/5	0/5
CrewAI	100%	16	1.0	1.0×	5/5	5/5
SmolAgents	60%	0.1	1.0	0.0×	1/5	1/5
OpenAI Agents	100%	33	1.0	1.1×	5/5	5/5
LlamaIndex	100%	8.8	1.0	0.0×	5/5	5/5
Agno	100%	47	1.0	0.0×	5/5	5/5

E Exploratory-Retrieval Suite: Per-Framework Results

Table 5 gives the per-framework results behind §5.5 of the main paper (five multi-document tasks, one run each).

Table 5: Unprompted exploratory retrieval: per-framework means over the five tasks.

Framework	Acc	S_{total} (MB)	D	Echo
LangGraph	100%	3.01	7.8	11.8×
AutoGen	47%	0.62	3.2	2.5×
InfiAgent	100%	3.00	1.7	6.7×
CrewAI	100%	0.68	2.2	2.8×
SmolAgents	67%	0.00	–	0.0×
OpenAI Agents	100%	0.37	1.4	1.4×
LlamaIndex	100%	0.93	3.1	3.1×
Agno	100%	1.17	1.8	2.0×

F Backend Replications

Full-roster replications (Qwen3.6-27B and Qwen2.5-7B). All eight frameworks re-ran the complete file-QA suite (10 tasks × 3 repetitions per backend; 480 fresh sandboxes) on two further backends served through the same provider channel as the main study. Under Qwen3.6-27B, per-framework S_{total} lands within 0.93–1.06× of the main study, accuracy is 98–100% throughout, and the persisting-framework

ranking is preserved at Spearman ρ=0.964 (spread 16.9× vs. 15.7×). Under the deliberately weak Qwen2.5-7B, task accuracy collapses for four frameworks (35–39%), and the per-answer records identify the mechanism: 80–94 of each collapsed framework's 150 answers are HTTP-400 errors from full-history prompts exceeding the endpoint's 32,768-token window as the 60 KB source files accumulate—context overflow, not answer quality. InfiAgent (windowed context) never hits the ceiling and holds 100%; LlamaIndex, whose default chat memory bounds the prompt while its store appends everything, holds 92%. Yet the storage ranking persists at ρ=0.893 (spread 12.0×): the retention signature is visible even when the model fails the task, which is precisely why storage must be read jointly with accuracy rather than alone—and the same design choice, whether history is windowed before reaching the prompt, drives both the storage class and the small-window robustness.

Table 6: Full-roster backend replications (mean per-task values; 30 runs per cell).

	Qwen3.6-27B				Qwen2.5-7B			
Framework	S_{total}	D	Echo	Acc.	S_{total}	D	Echo	Acc.
LangGraph	5.20	13.2	8.2×	100%	2.49	9.1	3.9×	35%
AutoGen	2.64	8.5	4.3×	98%	1.26	6.3	2.1×	39%
InfiAgent	2.13	1.8	1.7×	100%	2.14	1.8	1.7×	100%
Agno	1.16	2.5	0.8×	100%	0.77	2.0	0.6×	39%
LlamaIndex	0.94	4.2	1.3×	100%	1.20	5.5	1.7×	92%
OpenAI Agents	0.31	1.5	0.5×	100%	0.21	1.4	0.3×	35%
CrewAI	0.32	1.5	0.5×	100%	0.37	1.7	0.6×	81%
SmolAgents	0	–	–	99%	0	–	–	60%

Serving-channel note. One cell required a serving substitution: the sole API provider offering tool support for Qwen2.5-7B could not parse SmolAgents' zero-argument tool schema (the router returned "no endpoints found that support tool use" once that provider was excluded), so the SmolAgents×7B cell was served from the same weights on a self-hosted vLLM instance; all other cells used the shared API channel. Ten Qwen3.6-27B SmolAgents runs that hit the harness wall-clock limit during a provider-congestion window were treated as invalid and re-executed in fresh sandboxes under an extended limit, per the benchmark's task-identity rule.

Table 7: GPT-4o-mini subset (earlier three-framework, five-task protocol, superseded by the full-roster replications above; both columns use that protocol, so the DS value 5.27 is not comparable to main Table 1's ten-task 5.10). Rankings and D transfer; absolute bytes shift with model verbosity.

	File-QA S_{total} (MB)		D	
Framework	GPT	DS	GPT	DS
LangGraph	5.35	5.27	13.6	13.4
AutoGen	2.33	2.50	7.9	8.3
OpenAI Agents	0.33	0.33	1.6	1.6

GPT-4o-mini subset (earlier protocol).

G Growth Repetitions and Model Comparison

Repetition protocol. Every horizon of the three full-history frameworks (LangGraph, AutoGen, LlamaIndex) was rerun under two additional independent seeds (24 added runs; identical harness, backend, and judging), giving three runs per point; the remaining frameworks keep single runs. Table 8 reports per-horizon means. Two repetition cells failed the validity screen applied to every run (return code, accuracy band, store size within the sibling family) and were rerun; the failures coincided with an upstream serving-pool incident on the shared backend model (one harness-level timeout, runs with 20–34% of turns receiving HTTP-200 responses with empty content, one stalled connection). The final rerun enabled an environment-gated transport guard (90 s per-request timeout with client-side retries; re-issue of a request whose response arrives empty). The guard is inactive by default, was off for all other accepted runs, and does not alter persistence behavior—the session history stores only the final response of each call. The rejected attempts are preserved in the release archive (`longhorizon_runs/_anomalies/`): a manifest of all five attempts plus each attempt's measurement, answer, and adapter-log records where the failure mode produced them.

Between repetitions, scatter concentrates in absolute volume while the exponent is stable (per-seed fits in Table 8): LangGraph's T=100 point varies by ±10 MB yet its α moves by ±0.01—consistent with checkpoint-layer consolidation shifting *when* compaction lands rather than how volume scales.

Table 8: Per-horizon retention, mean ± SD over three seeded runs (MB); means and SDs are computed on unrounded per-seed fits. Per-seed full-range α fits: LangGraph $\{1.72, 1.75, 1.74\}$, AutoGen $\{1.76, 1.97, 2.12\}$, LlamaIndex $\{1.95, 1.77, 1.93\}$.

Framework	T=25	T=50	T=100	T=200
LangGraph	10.1±0.0	29.6±6.3	152.6±10.3	323.2±25.6
AutoGen	1.7±0.4	9.8±0.2	34.9±3.7	101.7±20.8
LlamaIndex	1.1±0.0	3.7±0.8	14.7±3.2	56.3±11.5

Model comparison. Table 9 compares power-law, linear, and quadratic fits over the four measured horizons. With four points and up to three parameters these comparisons are descriptive—the quadratic is near-saturated by construction—and the functional form is not identified at this n even with three runs per point; that is precisely why the main text claims superlinear growth over the measured range (via local slopes) rather than a validated scaling law. Archival policy is part of the configuration: LlamaIndex's latest-only footprint (newest snapshot per horizon) is 85, 163, 336, and 645 KB at $T \in \{25, 50, 100, 200\}$, fitting α=0.98 against 1.88±0.10 for the append-all series it ships by recipe—the growth class belongs to the archival configuration, not the framework in the abstract.

Table 9: Model comparison for retention growth. R^2 of power-law ($\log S_{\text{total}} \sim \alpha \log T$), linear ($S_{\text{total}} \sim a{+}bT$), and quadratic fits. α: mean $\pm$ SD of per-seed fits where three seeds exist, single-seed fit otherwise; R^2 columns are evaluated on the seed-1 series.

Framework	α	R^2 (power)	R^2 (linear)	R^2 (quadr.)
LangGraph	1.74±0.01	0.976	0.984	0.987
AutoGen	1.95±0.18	0.988	0.996	1.000
InfiAgent	1.15	0.999	0.996	1.000
Agno	0.98	1.000	1.000	1.000
LlamaIndex	1.88±0.10	1.000	0.967	1.000
OpenAI Agents	0.73	0.970	0.909	0.995
CrewAI	1.20	0.996	0.992	1.000

H Whole-Store Compression vs. Content Addressing

Table 10 makes the comparison of §7 of the main paper explicit. "zstd archive" is the opaque whole-store archive implied by main-paper Table 1 (S_{total} divided by C); "CAS store" is the schema-preserving compacted representation (main-paper Table 1, CAS column), which frameworks do not consume in place without a shim (§7 of the main paper); "CAS+zstd" archives the CAS output with the same zstd settings (measured on the same first-repetition runs). The opaque archive is always smallest, but nothing in it is usable without full decompression. Where redundancy is exact repetition (AutoGen's event log, Agno's session rows), archiving the CAS output matches the opaque archive within measurement noise. Where redundancy is *near-duplicate*—LangGraph's evolving checkpoints differ slightly at each turn—exact-match content addressing keeps the variants as distinct blobs and the cold archive pays up to $10\times$ over whole-store zstd; closing that gap needs delta encoding, which we leave to future work.

Table 10: Per-task retained MB under four representations (file-QA suite). "zstd archive" is derived from main-paper Table 1's C (stream-concatenated compression bound, not a restorable archive); "CAS+zstd" is measured on a real tar stream.

Framework	Default	zstd archive	CAS store	CAS+zstd
LangGraph	5.10	0.038	0.50	0.389
AutoGen	2.49	0.023	0.07	0.025
InfiAgent	2.12	0.049	0.36	0.177
Agno	1.15	0.027	0.06	0.027
LlamaIndex	0.93	0.023	0.09	0.054
OpenAI Agents	0.33	0.024	0.07	0.039
CrewAI	0.32	0.023	0.05	0.044

I Conversation-Reconstruction Protocol

The $R{=}3$ validation of §5.4 of the main paper uses an HTTP logging proxy interposed between each adapter and the model provider; it records every request body verbatim and is the ground truth. Each of the six $R{=}3$ frameworks ran three proxied tasks (18 proxied runs; 194 recorded calls in total, with per-task totals of 65, 64, and 65 across the six frameworks). A per-framework extractor then rebuilds the conversation from retained bytes *alone*: LangGraph from checkpoint snapshots; AutoGen from saved per-turn state (normalizing its thought-field encoding); OpenAI Agents from session rows (merging text and function-call items belonging to one assistant turn); LlamaIndex from the serialized context; Agno from its session-run rows; InfiAgent from its raw trace channel. The scoring contract is *contiguous-subsequence* matching: every recorded request's message list must appear, field-by-field (role, content, tool-call identifiers, arguments, tool results, order), as a contiguous subsequence of the history rebuilt from retention. This contract is what makes windowed senders comparable: Agno transmits a bounded window of past runs per request, and the window it sent is exactly what must match. All 194 recorded calls satisfy the contract (per-call archives for all 18 runs ship in the artifact). System-prompt retention is reported separately in the main text because three frameworks keep the system prompt in code-side configuration rather than in any store. Scope: R measures exact reconstructability of the ordered conversation content associated with each call, not byte-identical reconstruction of the complete provider request; call boundaries during verification come from the recorded requests, and the resume probe (Appendix J) shows the stores support continuation, not that call boundaries are self-describing. Per-call reconstruction diffs ship in the supplementary archive.

J Process-Boundary Resume Probe

The reconstructability grade asks whether retained bytes can reconstruct history; this probe asks whether they can *continue* it. Phase A runs each framework's unmodified benchmark adapter in its documented durable-session configuration: the agent is told a nonce registry code, acknowledges it, and the process exits (the crash point is between turns). Phase B starts a fresh process on the same sandbox home, reattaches to the same documented session identity, and asks for the code; workspace files do not contain it, so only the retained store can supply it. Because the AutoGen and LlamaIndex adapters persist state without ever reloading it, their phase B exercises the frameworks' official restore APIs (`load_state`, `Context.from_dict`) in the fresh process.

Table 11: Process-boundary resume: can a fresh process continue the session from retained bytes alone?

Framework	R	Resumed	Phase-B mechanism
LangGraph	3	yes	`thread_id` + SQLite checkpointer
AutoGen	3	yes	`load_state` of saved state
OpenAI Agents	3	yes	`SQLiteSession`
LlamaIndex	3	yes	`Context.from_dict`
Agno	3	yes	`session_id` + session db
InfiAgent	3	yes	task identity + task store
CrewAI	1	no	task-output store; no session API
SmolAgents	0	–	nothing retained

All six $R{=}3$ frameworks return the exact code from retention alone; CrewAI answers that it has no record of the

Table 12: Meter calibration. Five copies per block except small-lines (eight). "part": under ≥doubly-nested encoding only probes free of escapable characters match, so echo is a lower bound there.

Scenario	Encoding	D_{true}	D	naive D	echo_{exp}	echo
single-esc	JSONL, `json.dumps`	5.0	5.0	2.1	5.0	5.0
sqlite-exact	SQLite text cells	5.0	5.0	1.6	5.0	5.0
double-esc	JSON inside JSON	5.0	5.0	2.4	part	1.7
triple-esc	three nesting levels	5.0	5.0	2.1	part	1.7
near-dup	1% byte edits per copy	≈1	1.0	1.0	–	2.0
zlib-prefix	compressed growing prefixes	≈1	1.1	1.0	0	0.0
small-lines	400-char lines ×8	8.0	8.0	1.0	–	8.0

code; SmolAgents has nothing to resume by construction. The grade therefore stratifies process-boundary session resumability exactly on this roster. Scope: the probe crashes between turns and resumes conversation state; recovery of mid-turn tool state or workflow-graph position is a stronger capability we do not test.

K Meter Calibration on Known-Duplication Stores

We calibrate the meter on synthetic stores whose duplication is known by construction (generator script and stores ship in the artifact). A 40 KB source file—the probe source—is cut into ten 4 KB blocks; each scenario persists those blocks with a known copy count under a different encoding. "Naive D" runs the same content-defined chunking over raw file bytes instead of logical streams—the measurement §3.3 of the main paper argues against.

Three observations. (i) D is exact on every exact-duplication scenario, including duplicates below the chunker's minimum size (whole-stream hashing), while naive raw-byte chunking underreports the same stores at 1.6–2.4×—the serialization-masking effect of §3.3 of the main paper reproduced on known ground truth. (ii) Echo is exact through single JSON escaping, the encoding real frameworks apply to message content, and degrades to a lower bound under doubly-nested encodings, where only probes without escapable characters keep matching. (iii) The designed boundaries hold: near-duplicate copies and compression-wrapped redundancy are invisible to exact-match analysis (D≈1) while C still flags the former (17.7×)—the same signature that separates LangGraph's evolving checkpoints in Appendix H.

L Fixed-Trace Protocol: First Instantiation

Per-framework results behind §5.6 of the main paper. The mock endpoint adapts tool schemas and termination contracts per framework (InfiAgent terminates via its `final_output` contract); every framework persists the same logical content—three reads, 182.7 KB of observations, one answer.

Two observations. First, identical-content encoding amplification spans 1.03× (LlamaIndex: this single-question trace serializes one context snapshot, consistent with the latest-only analysis of Appendix G) to 6.86× (InfiAgent's multi-channel stores)—a 6.7× spread carrying no agent-policy contribution. Second, transport shape itself matters for per-step checkpointers: under a sequential-transport variant (one tool call per model turn), LangGraph rises from 3.24× to 7.40×—one checkpoint per superstep (variant run archived in the artifact)—so the protocol pins the batched transport and discloses the variant.

Table 13: Fixed-trace results (batched transport). "×/obs" = retained bytes per observation byte. [a]CrewAI executes the full trace but paraphrases the scripted final answer, failing exact match; retention covers the completed trace. SmolAgents retains nothing and is out of scope.

Framework	Trace	Answer	S_{total} (KB)	×/obs
InfiAgent	✓	✓	1,254	6.86
CrewAI[a]	✓	–	624	3.42
LangGraph	✓	✓	592	3.24
AutoGen	✓	✓	555	3.04
Agno	✓	✓	404	2.21
OpenAI Agents	✓	✓	212	1.16
LlamaIndex	✓	✓	188	1.03

M Continuous Shared-Store Protocol

Fresh sandboxes maximize attribution but leave open whether a long-lived shared backend amortizes per-task cost. This protocol runs ten identical fixed-trace tasks against one sandbox and one documented session identity per framework, measuring cumulative retention after each task; marginal cost is $\Delta S(k) = S(k) - S(k{-}1)$.

Table 14: Ten identical tasks on one shared store (mock endpoint; all answers correct for every included framework).

Framework	$S(1)$	$S(10)$	$\Delta S(1)$	$\Delta S(10)$	Shape
LangGraph	592 KB	95.2 MB	592 KB	18.5 MB	rising
Agno	404 KB	7.0 MB	404 KB	748 KB	mildly rising
AutoGen	555 KB	3.8 MB	555 KB	371 KB	flat
OpenAI Agents	212 KB	1.8 MB	212 KB	184 KB	flat
LlamaIndex	188 KB	188 KB	188 KB	0	constant

The amortization hypothesis *inverts* for full-history checkpointers: on one shared thread, LangGraph's ten identical tasks retain 95.2 MB versus 5.8 MB as ten fresh sandboxes—the fresh-sandbox protocol *understates* long-lived deployment cost for such designs rather than overstating it, because every new task re-checkpoints the whole accumulated history. Overwrite and append-once designs amortize as expected (LlamaIndex's per-task snapshot is overwritten; AutoGen's snapshots overwrite while its event log appends). InfiAgent is excluded by its own documented contract: task identities must not be reused across invocations—the rule our harness enforces (§4.3 of the main paper)—and driving ten tasks through one identity reproduces exactly the contamination its documentation warns about (runaway rounds to its 60-round cap).

N Meter Implementation Audit and Threshold Sensitivity

Delta accounting. The runs in this paper were captured with a size-inventory baseline: a pre-existing file was treated as unchanged when its size was unchanged, and a changed file was counted at its full post-run size. Both approximations were audited over all 509 sandboxes against the definition of §3.1 of the main paper. (i) Pre-existing bytes inside changed baseline files total 31,952 B against an aggregate S_{total} of 1,404.7 MB—0.0022%, worst single run 0.056%—so full-size versus increment attribution does not move any reported number. (ii) Same-size in-place rewrites: every baseline workspace file was byte-compared against its pristine task corpus source; zero modifications across all 509 runs. (iii) Only InfiAgent populates the home directory before the run (its setup scaffolding); its same-size baseline files were verified separately—300 corpus copies byte-identical to their pristine sources and 150 configuration files byte-identical across repetition seeds—leaving no evidence of any undetected rewrite. The released meter additionally hashes every baseline file (SHA-256) and reports an increment-based `S_total_delta` alongside the published accounting, closing this gap by construction.

Stream-length threshold. Streams shorter than 64 B (SQLite cells, JSONL lines) are excluded from both the numerator and denominator of D; numeric and NULL cells are never content bytes and are excluded at any threshold. Recomputing D on the file-QA first-repetition subset at thresholds of 0, 16, 32, 64, and 128 B changes no framework's value by more than 0.1 (LangGraph 13.3–13.4; all others flat to one decimal), so the threshold does not drive any conclusion.

Setup staging and boundary. The baseline snapshot is taken after framework-side setup, so bytes a framework stages *before* execution are excluded from S_{total}. Only InfiAgent stages: its setup copies the task corpus into its managed task directory (0.60 MB mean on file-QA). Under a total-operational view (staging + retained), its file-QA mean rises from 2.12 to 2.72 MB—moving it above AutoGen (2.49 MB) in absolute terms—while the 15.7× LangGraph/CrewAI headline is unchanged. We keep staged copies of immutable task inputs out of S_{total} because they duplicate provided inputs rather than execution history; both accountings ship in the artifact. Sizes are apparent file lengths (`st_size`), not allocated blocks; sparse-file and copy-on-write divergence is second-order at these scales. The redirected boundary does not cover system temp directories or remote telemetry; a full-filesystem container audit is the extension path for adversarial submissions.

O Automated R Scorer: Construct Validation

The automated R rubric is a *screening* instrument: per-call structure is identified from per-framework channel signatures (audited once per adapter), `.jsonl` channels must additionally parse as JSON-object records, and completeness uses whole-line probes. Order, role, argument, and boundary fidelity are established by the reconstruction study of §5.4 of the main paper and the resume probe of Appendix J—not by the automated score. To make the division of labor explicit we scored seven constructed stores with known defects (Table 15).

Table 15: Construct check of the automated R scorer on synthetic stores built from a real task corpus. "3*": content-complete per-call stores with order or argument defects receive an automated 3 by design; those properties are verified by reconstruction, which compares roles, contents, tool-call ids, arguments, and order field-by-field.

Constructed store	Intended	Auto R	Verdict
True per-call record	3	3	correct
Aggregate blob, spoofed name	1	1	correct
Aggregate blob, neutral name	1	1	correct
Per-call, shuffled order	3*	3	insensitive*
Per-call, missing tool args	3*	3	insensitive*
Per-call, truncated observations	2	2	correct
Per-call, summaries only	2	2	correct

An earlier version of the scorer accepted the spoofed-name aggregate blob (filename signature alone); the released version adds the JSON-record gate, which corrects that case and leaves every published framework grade unchanged. Framework-level grades thus rest on the validated adapter contract plus one stated assumption: serialization semantics are invariant across tasks at a fixed framework version and configuration. For third-party submissions this generalizes cleanly: extractors need not be trusted, because the logging proxy is the trust anchor—a submitted (or even machine-generated) extractor is verified mechanically, field-by-field, against recorded requests on hidden-seed tasks, so parsing is crowdsourced while verification stays sound.

P Production Deployment Measurement

With the operator's authorization we measured a production InfiAgent deployment (supply-chain quoting). Collection was whitelist-based and irreversibly de-identified at the source: only byte counts, file counts, channel categories, coarsened months, and tool-name frequencies were exported—no message content, file names, user identifiers, or business data left the environment (the collector ships in the artifact; tool names are pseudonymized in the released aggregates).

Deployment root. Eight tasks, 442 MB. Channel composition: raw traces 148.0 MB, service logs 98.5 MB, task stores 71.6 MB, other 69.4 MB, debug logs 44.1 MB, conversation stores 4.8 MB, configuration 0.3 MB; the residual ∼5.3 MB (1.2%) is files outside the collector's channel taxonomy. Trace, log, and debug channels hold 290.6 of the root's 442 MB (66%) while conversation stores hold 1%, amplifying the controlled composition finding (Appendix B) at production scale. Per-task retention: median 715 KB, P90 81.4 MB—a heavy tail two orders of magnitude above the median.

Deployment-wide scale. As of July 2026 the operator reports ∼56 GB of accumulated retained data across ∼1,000

recorded agent rounds for this deployment—an operator-attested aggregate over the full store, of which the eight-task, 442 MB root above is the collector-audited subset. "Round" is the deployment's unit of work and is not calibrated to our task definition, so we derive no per-task mean from it; the figure establishes practical scale in one deployment, not population prevalence.

Single-task decomposition. One complete quoting task retains 131 MB: a staged input workbook (42.4 MB, in the framework's hidden upload area), framework residue of 79.2 MB (raw trace 47.7, runtime logs 29.7, action log 1.5, history database 0.3), staged reference material (6.8 MB), and 2.6 MB of delivered artifacts. Residue outweighs deliverables $\sim$30$\times$ with real work products—the absolute-scale counterpart of the write-suite ratios (§7 of the main paper)—and input staging (49.2 MB) mirrors the boundary of Appendix N.

Workload profile. The task issued 184 model calls; tool-name frequencies are dominated by profile updates (219), program steps (62), resource lookups (50), and structured reads (32), with file writes rare—direct measured evidence for the read-dominant premise of boundary (i) in §8 of the main paper. The de-identified aggregates ship in the supplementary archive.

Q History-Retention Channel Decomposition (Executed)

The system track measures documented configurations that support different capabilities (§4.2 of the main paper). This decomposition measures how much of the default spread is associated with channels beyond conversation-history retention, by *executing* each configuration's minimal history-retaining form on the full file-QA suite (10 tasks, single repetition, main-study backend). Three forms required a change and were re-run end-to-end: LlamaIndex latest-only archival (delete the previous snapshot after each write), AutoGen events-off (the documented switch: no event logger, state channel kept), and LangGraph newest-checkpoint pruning (an adapter-level wrapper deleting superseded checkpoint rows, then `VACUUM`). Functional equality is verified on all 30 runs: accuracy is unchanged (150/150 questions correct) and every retained store still scores R=3. OpenAI Agents and Agno are natively history-only (their main-study runs are the executed evidence); InfiAgent's channels expose no switch, so it remains a disclosed post-hoc split. This remains a decomposition of the default spread, not a capability-normalized efficiency ranking: the executed forms match on history reconstruction, not on every capability a default bundle serves.

Table 16: History-only forms on file-QA (mean over 10 executed tasks where re-run). Source: executed (re-run end-to-end, accuracy and R verified), native (default already history-only; main-study runs), split (post-hoc channel exclusion; channels not switchable).

Configuration	MB	Source
LlamaIndex (latest-only archival)	0.31±0.00	executed
LangGraph (newest checkpoint)	0.31±0.03	executed
OpenAI Agents (session)	0.33±0.00	native
AutoGen (state, events off)	0.90±0.00	executed
Agno (session db)	1.15±0.00	native
InfiAgent (conversation channel)	1.20	split

The decomposition brackets the headline: the default spread is $15.7\times$; restricted to history retention it is $3.9\times$, with three unrelated codebases converging within 7% (0.31–0.33 MB)—storing the same conversation costs nearly the same bytes once the capability bundle is stripped. The residual $3.9\times$ is history-encoding overhead, consistent in magnitude with the fixed-trace encoding spread (§5.6 of the main paper). Most of the default-configuration dispersion is therefore *what defaults choose to keep*, not irreducible storage inefficiency—which is exactly what a default-experience benchmark should surface.

Table 17: Persisted objects and recovery capabilities. Resume is measured here (Appendix J); Wf. (workflow-state recovery) and Ev. (event/debug provenance) are vendor-documentation claims, untested.

Framework	Persisted objects	Resume	Wf.	Ev.
LangGraph	graph checkpoints	yes	yes	–
AutoGen	state snaps + events	yes	partial	yes
OpenAI Agents	message session	yes	–	–
LlamaIndex	serialized context	yes	partial	–
Agno	session runs (window)	yes	–	–
InfiAgent	task store + traces	yes	yes	yes
CrewAI	task outputs	no	–	–
SmolAgents	none	no	–	–

R Capability Matrix (Documentation-Based)

Table 17 summarizes what each documented durable-session configuration persists and which recovery capabilities it offers, per vendor documentation; "resume" is the process-boundary probe of Appendix J (measured), the other columns are documentation claims we did not independently test. This is the boundary behind §4.2 of the main paper: equal-R frameworks are comparable on history reconstruction, not on every column below.

S Reconstructability–Retention Figure

Figure 2 plots each framework's default retention against its post-CAS retention, annotated with R and accuracy; the reading is discussed in §5.4 of the main paper.

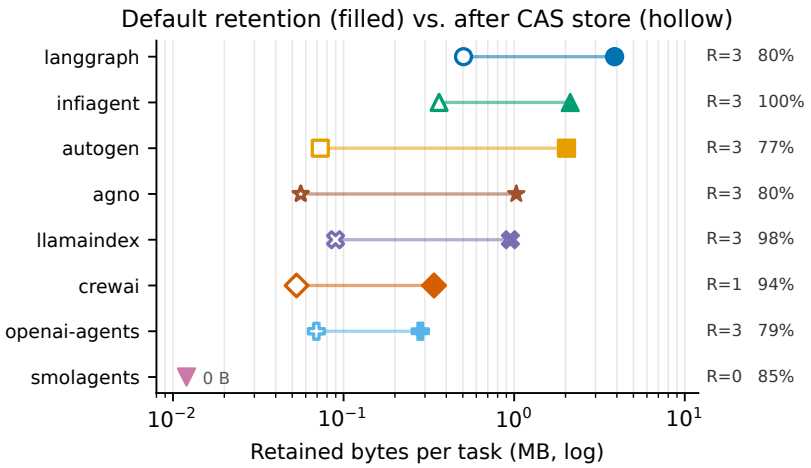

Figure 2: Reconstructability versus retention (discussed in §5.4 of the main paper). Filled: default retention per task; hollow: the same runs after the content-addressed store; right margin: R and accuracy. R=3 is obtained at 0.33–5.10 MB under defaults and 0.06–0.50 MB after content addressing (the 0.05 MB post-CAS point is CrewAI, R=1).